Assuring Dependable Cloud-Based System Engineering: A Cloud Accountability Method

Assuring Dependable Cloud-Based System Engineering: A Cloud Accountability Method

David Ebo Adjepon-Yamoah
Centre for Software Reliability
School of Computing Science
Newcastle University
Newcastle-upon-Tyne NE1 7RU, UK
Email: *d.e.adjepon-Yamoah@ncl.ac.uk*

Zhenyu Wen
Centre for Intelligent Systems and their Applications
School of Informatics
University of Edinburgh
Edinburgh EH1 2QL, UK
Email: *zwen@inf.ed.ac.uk*

***Abstract*—This work introduces a methodology for cloud accountability that assures system dependability in terms of *availability* and *reliability*. This assurance is provided relative to the cloud service level agreement. The presented methodology is guided by the NIST SP800-86 digital forensic model, that motivates the *collection*, *examination* and *analysis* of data from the cloud platform, and the generated *evidence* including logs and context are *reported* to appropriate cloud *agents*. As part of this work, we present a novel approach to collecting digital evidence to support cloud-based system dependability, using the Virtual Machine Introspection (VMI) technique. Our VMI approach complements, as well as checks the dependability metrics provided by the cloud service providers (CSPs) as evidence. This methodology, including the VMI approach is particularly relevant since it provides a means of addressing the perceived lack of *trust* for cloud-based services towards *cloud accountability*.**

Our research focuses on applying an *evidence-based* methodology - *cloud accountability method* - to cloud-based system engineering for assuring cloud *agents* of the dependability of cloud platforms.

I. Introduction

The cloud computing technology of today and the future promises to bring demonstrable benefits to people's lives [1]. It is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. This outsourcing model is attractive for businesses that wish to minimize their computing and storage infrastructure cost [1]. Other appealing features are the multi-user capability, flexibility, speed, scalability, etc. Scalability is a key attribute of cloud computing and is achieved through server virtualisation [2]. However, the responsibility of the cloud service providers (CSPs) is often called to question. Literature review shows that these concerns are largely related to the assurance of security [3], [4] and dependability [5], [6], [7] on the cloud.

With the movement of software engineering from local computers to the cloud, software developers need to be assured of the dependability of the engineering support deployed to the cloud. The predefined and mutually agreed upon service level agreement (SLA) provided by CSPs attempt to assure developers of cloud dependability. However, due to the cloud platform's inherent complexity and large scale, production cloud computing systems are prone to various run-time problems caused by hardware and software faults, cloud run-time management decisions and environmental factors [7], [8]. Cloud *agents* such as system developers, CSPs, and cloud regulators need to be informed about possible or actual violations of the SLAs. A robust mechanism is needed for violation detection, notification, logging and resolution. Once the cause of the violation is found, each violator is regarded as being accountable for their fault.

This however highlights the need for cloud accountability [9], [10]. Creating accountability in the cloud is seen as a solution to users' lack of trust [9]. In cloud accountability, the cloud *agents* are able to check whether the cloud is running the service as agreed. If a problem appears, they should be able to determine which of them is responsible, and to prove the presence of the problem to a third party, such as an *arbitrator* or a *judge* [11]. Such an activity should be based on *evidence* from an evidence-based auditing process.

The main contribution of this work is to introduce a methodology for cloud accountability that assures system dependability in terms of availability and reliability. This assurance is provided relative to the cloud SLA. The presented methodology is guided by the NIST SP800-86 forensic model [12], that motivates the *collection*, *examination* and *analysis* of data from the cloud infrastructure, and the generated *evidence* including logs and context are *reported* to appropriate cloud *agents*. This work also presents a novel approach for collecting digital evidence to support cloud-based system dependability, using the *Virtual Machine Introspection* (VMI) [13] technique.

This methodology aims to: (1) assure cloud *agents* of the dependability of the cloud platform with reference to cloud SLAs, (2) quantify SLA violations, and (3) serve as an evidence-based benchmark for choosing a relatively dependable cloud provider.

The broad research question (**RQ**) that we seek to answer is "*Can a cloud accountability method be used to meaningfully assure availability and reliability of deployed systems, relative to the cloud platform's service level agreement (SLA)?*". To answer this question, we validate the hypothesis (**H1**): "*The cloud accountability method can be used to meaningfully assure the availability and reliability of cloud-based systems*".

II. Background and Related Work

In this section, we discuss some dependability attributes of systems deployed to the cloud. The SLA provided by Amazon Web Services (AWS) CSP is also discussed. Afterwards, an evidence-based approach motivated by data forensic for cloud accountability is presented. Finally, the Net Present Value (NPV) approach, and related work are introduced.

A. Dependability Attributes of Software Systems

In order to work on dependability of cloud computing systems, we must identify the major attributes [14] which can quantify the dependability of system in different perspectives. Some very important attributes for such systems include availability, reliability, performance, security and recoverability [7].

B. Cloud Service Level Agreement

The assessment of the Amazon Web Services' (AWS) SLA for their Elastic Cloud Compute (EC2) service [15] only guarantees two things: (1) that the EC2 API will be available to allow for the launching of new instances 99.9% of the time, and (2) that more than 0% of user instances will be able to access the Internet 99.9% of the time (i.e. it is an outage if 100% of user instances cannot reach the Internet 99.9% of the time). The AWS SLA does not explicitly cover the reliability of the AWS EC2 instances. Even with the availability assurances for the AWS EC2 instances, they are rather vague. That said, the AWS SLA provides some grounds for assuring system availability.

C. Digital Forensic Approach for Cloud Accountability

Digital forensic is obtaining, preserving, analysing, and documenting digital evidence from digital devices [16]. This process has been widely accepted and largely used in investigating cloud security issues [17], [18]. This activity is called cloud forensic, which is defined broadly by [19] as "an application of digital forensic science in cloud computing environments". Some evidence sources for cloud forensic are:

1) Hypervisors: Hypervisors or virtual machine monitors (VMMs), such as Citrix's Xen, VMWare ESXi and Microsoft HyperV, are used to manage virtual machines (VMs) and their various hardware resources [20]. It can provide runtime statistics, but also information can be derived from the hypervisor using advanced techniques like VMI[13]. VMI is the technique of locating and accessing the digital forensic evidence on a running VM (user-VM) from another isolated running VM (admin-VM) which is co-located on the same hardware and which has required privileges to access the hypervisor layer. VMI is transparent and does not interrupt the work-flow of the target user-VM nor can it be detected from there. VMI is especially of interest for security-related techniques, e.g. intrusion detection.

2) Cloud Management System (CMS): CMS is a huge source for evidence information. It is the central controlling component of a cloud infrastructure and provides information about user logins, cloud service usage, access rights, configuration, resource provisioning, policies, location, etc. CSPs like AWS provide the AWS CloudWatch with application programming interface (API) for such information gathering.

D. Net Present Value

The idea of Net Present Value (NPV) encompasses the concept of time value of money and takes into consideration that money spent or obtained in future periods will have a different value than money spent or obtained in the present [21]. For decades, NPV has been a standard method for the financial appraisal of projects. NPV calculations can be currently found in every project document (business case, project plan, etc.) and project managers throughout the world use this methodology to compare the value of different projects against investment targets [22]. In the same light, users of cloud resources should be able to know the value of their cloud investments relative to their investment targets.

E. Related Work

The Cloud Broker Architecture for Dependability [5], Phantom [6], and the A4Cloud FP7 Project [23] are very similar conceptually to this work. The Phantom [6] is most similar in its operations. However, DBA [5] is concerned with fault detection, fault evaluation and taking decision for recovery or migration as a means for assuring dependability. The main shortcoming of DBA (i.e. poor memory management) is inherited from the CORBA platform [24]. This leads to ad-hoc solutions with regards to avoiding memory leaks, which then introduces large overhead cost. The A4Cloud FP7 Project [23] on the other hand, is focused on cloud accountability in terms of data security, which is a departure from our aim.

A major distinction of our work is in the use of the VMI technique for digital evidence collection, and the Reactive Middleware (RM) for managing dependability violations. By moving the monitoring tool outside the focused VM and leveraging function-call injection techniques using VMI, we will be able to minimise operational cost for collecting and monitoring digital evidence. Also, the RM notifies cloud *agents* of violations, and afford them the option of managing these violations. All violations, activities in response to the violations, and responsible agents are documented as *logs*. These logs can be used as (1) *evidence to claim compensation*, and also to (2) *serve as a source of data for predicting dependability violations* using a form of machine learning.

III. Cloud Accountability Method

We introduce a cloud accountability method (CAM) that is guided by a digital forensics model. The model is based on the well-established and widely accepted work of NIST SP800-86 [12]. The NIST SP800-86 guide shows how digital forensics can support *incident handling*. The forensic process comprises the following phases:

1) Collection: The first phase in the process is to identify, label, record, and acquire data from the possible sources of relevant data, while following guidelines and procedures that preserve the integrity of the data. Collection is typically performed in a timely manner because of the likelihood of losing dynamic data such as current network connections.

2) Examination: It involves forensically processing large amounts of collected data using a combination of automated and manual methods to assess and extract data of particular interest, while preserving the integrity of the data.

3) Analysis: The next phase of the process is to analyse the results of the examination, using legally justifiable methods and techniques, to derive useful information that addresses the questions that were the impetus for performing the collection and examination.

4) Reporting: The final phase is reporting the results of the analysis, which may include describing the actions used, explaining how tools and procedures were selected, determining other actions to be performed, and providing recommendations for improvement to policies, guidelines, procedures, tools, and other aspects of the forensic process. The formality of the reporting step varies greatly depending on the situation.

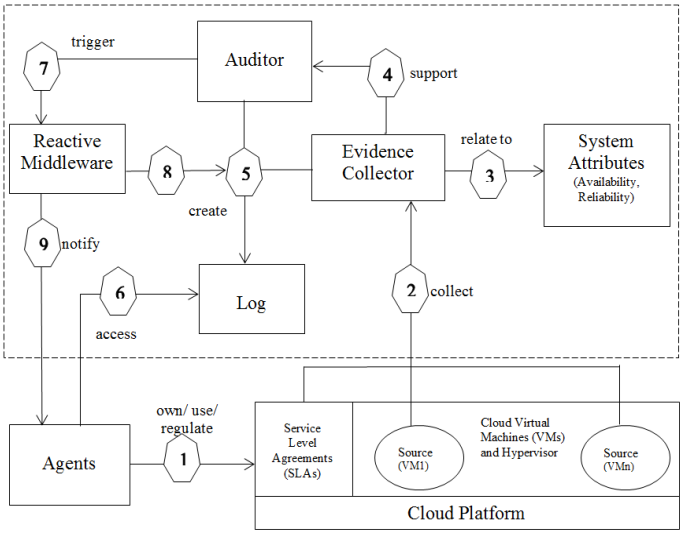

Fig. 1. Conceptual Model of the Cloud Accountability System

A. Cloud Accountability System

The Cloud Accountability System (CAS) (see Figure 1) is constituted by four main components with connectors. These components are the *Evidence Collector*, *Auditor*, *Reactive Middleware*, and a *Logging System*. The functions of these components are briefly introduced.

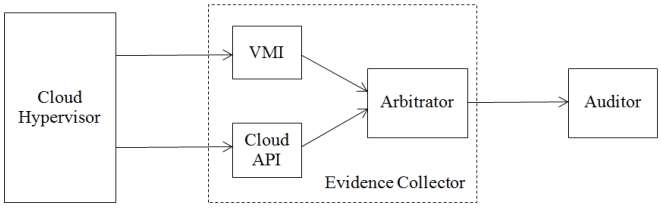

Fig. 2. Evidence Collector

1) Evidence Collector (EC): This system collects availability and reliability metrics from the VM hypervisor using two main approaches; Virtual Machine Introspection (VMI) technique and the Cloud Management System (CMS). The VMI uses a VMI library in C language, and Python scripts to collect these metrics. Also, the CMS uses Amazon Web Services (AWS) API. Figure 2 provides an overview of the Evidence Collector system. The Arbitrator sub-system reconciles metrics to support data integrity. All activities are saved as logs.

2) Auditor: The Auditor computes the *means* of the set of metrics, and then aggregates them. The aggregated metrics are then compared to the cloud platform's SLA and a call is triggered to the Reactive Middleware if there are violations. Also, the Auditor computes the Net Present Value (NPV) of the investments that cloud agents (i.e. system developers) have made subject to the recorded dependability violations. The system also derives the monetary value of the dependability violations to provide a clearer understanding to the investor (i.e. system developers). All activities are saved as logs.

3) Reactive Middleware (RM): The RM helps to manage notifications. The RM notifies system agents of dependability violations, and afford them the option of managing these violations. All violations, activities in response to the violations, and responsible agents are documented as *logs*. These logs can be used as evidence to claim compensation (i.e. "Service Credits"), and also serve as a source of data for predicting dependability violations using a form of machine learning.

4) Logging System (LS): In this system, all the activities of the CAS components are documented and saved as logs. This where all the generated *evidence* are stored.

The CAS components are implemented to measure in runtime the metrics that are related to the architecture availability and reliability. Such metrics are *Mean Time to Failure (MTTF)* or *Mean Time before Failure (MTBF)*, *Mean Time to Repair (MTTR)*, *system operation time*, *number of time periods*, etc. From Figure 1, the data gathered are compared with those from the cloud platform's API and the SLA (i.e. see *Steps 1, 2, 3, 4*), and then *notifications* are triggered to cloud agents if there are dependability violations (i.e. see *Steps 5, 6, 7, 8, 9*).

B. Steps of Methodology

The steps of the cloud accountability method are below;

1) The *Evidence Collector* (EC) is assigned to a set of software Systems [S_N] deployed on virtual machines (VMs) of the cloud.
2) The EC gathers dependability metrics (for availability, [A] and reliability, [R]) from their respective [S_N].
3) The metrics collection is synchronised among [S_N] at a constant time interval, t_i (i is initialised to zero).
4) The gathered dependability metrics are sorted and examined for data integrity by the *Arbitrator* (Figure 2).
5) The set [A, R]$_N$ is sent to the *Auditor* using synchronous procedure calls.
6) The *Auditor* classifies [A_N] and [R_N] for [S_N].
7) The *Auditor* then computes the *means* of [A_N] and [R_N] respectively.
   - Mean for Availability, $[mA]_N = \frac{\sum[A_N]}{N}$

- Mean for Reliability, $[mR]_{\mathrm{N}} = \dfrac{\sum[R_{\mathrm{N}}]}{N}$

8) The *Auditor* compares $[\mathrm{mA}]_{\mathrm{N}}$ and $[\mathrm{mR}]_{\mathrm{N}}$ to the corresponding data ($[\mathrm{cA}]_{\mathrm{N}}$ and $[\mathrm{cR}]_{\mathrm{N}}$) from the cloud's CMS for a *reasonable margin of error* (i.e. acceptable metric range, $[\mathrm{AMR}]_{\mathrm{c}}^{\mathrm{m}}$), based on the SLA of the cloud instance, and reliability benchmark.
9) If any data/metric of the two sets of data from (8) *violates* the SLA's $[\mathrm{AMR}]_{\mathrm{c}}^{\mathrm{m}}$, the *Auditor* triggers a call to the *Reactive Middleware* (RM).
10) The RM then sends a *notification* to the cloud *agents*.
11) All activities and cloud *agents* involved are *logged*.
12) The RM provides options such as requesting for "Service Credits" from the CSP using the *log* from (11).
13) The *Auditor* also computes the Net Present Value (NPV) of the virtual instance with respect to the identified means from (7), and/or violations from (9).
14) The monetary value [MVAR] of the *means* from (7) is derived.
15) If there was no violation from (9), an analysis of the comparison in (8), the net present value (NPV) from (13), and the [MVAR] are saved as a *log*, l_{N}.
16) This log, l_{N} is sent as a *notification* to the cloud *agents* as periodic reports (e.g. weekly, monthly).
17) The method continues in a loop at (3) for time, t_i*+5000ms* if none of the assigned VMs in (1) failed.
18) If there is any recorded failure, the method will continue in a loop at (1) for time t_i*+5000ms*.

The steps of the presented cloud accountability method are briefly classified according to the forensic process phases of the NIST SP800-86 guide:

- **Collection**: Steps 1, 2, 3, 5, 17 and 18
- **Examination**: Steps 4 and 6
- **Analysis**: Steps 7, 8, 9, 12, 13, and 14
- **Reporting**: Steps 10, 11, 15, and 16

IV. Conclusion

We evaluate the cloud accountability method by designing a *scenario* for a cloud-based Reactive Architecture that supports system engineering, based on Figure 1 *model*. The *scenario* is analysed using Continuous Time Markov Chains (CTMCs). CTMCs are able to capture system behaviour at the virtual machine (VM) level. At this point, we answer the research question (**RQ**) by validating the hypothesis (**H1**). In our future work, an implementation based on the *model* will be developed to collect, examine and analyse data from cloud platforms. Also, the analysis will be reported to cloud *agents*, for the purpose of assuring the accountability of cloud services.

References

[1] X. Zheng, H. Ye, C. Tang, C. Rong, and G. Chen, "A Survey on Cloud Accountability," in *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*, Dec 2013, pp. 627–632.

[2] Y. Jadeja and K. Modi, "Cloud computing - concepts, architecture and challenges," in *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, March 2012, pp. 877–880.

[3] A. Hendre and K. Joshi, "A Semantic Approach to Cloud Security and Compliance," in *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, June 2015, pp. 1081–1084.

[4] M. Ramachandran and V. Chang, "Recommendations and Best Practices for Cloud Enterprise Security," in *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, Dec 2014, pp. 983–988.

[5] W. Abderrahim and Z. Choukair, "Trust Assurance in Cloud Services with the Cloud Broker Architecture for Dependability," in *High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on*, Aug 2015, pp. 778–781.

[6] B. Hadley, A. Hume, R. Lindberg, and K. Obraczka, "Phantom of the cloud: Towards improved cloud availability and dependability," in *Cloud Networking (CloudNet), 2015 IEEE 4th International Conference on*, Oct 2015, pp. 14–19.

[7] Y. Pan and N. Hu, "Research on Dependability of Cloud Computing Systems," in *Reliability, Maintainability and Safety (ICRMS), 2014 International Conference on*, Aug 2014, pp. 435–439.

[8] J. Zhou, Z. Chen, J. Wang, Z. Zheng, and W. Dong, "A Runtime Verification Based Trace-Oriented Monitoring Framework for Cloud Systems," in *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*, Nov 2014, pp. 152–155.

[9] S. Pearson, "Toward Accountability in the Cloud," *Internet Computing, IEEE*, vol. 15, no. 4, pp. 64–69, July 2011.

[10] J. Yao, S. Chen, C. Wang, D. Levy, and J. Zic, "Accountability as a Service for the Cloud," in *Services Computing (SCC), 2010 IEEE International Conference on*, July 2010, pp. 81–88.

[11] A. Haeberlen, "A Case for the Accountable Cloud," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 2, pp. 52–57, Apr. 2010.

[12] K. Kent, S. Chevalier, T. Grance, and H. Dang, "SP 800-86. Guide to Integrating Forensic Techniques into Incident Response," Gaithersburg, MD, United States, Tech. Rep., 2006.

[13] R. Poisel, E. Malzer, and S. Tjoa, "Evidence and Cloud Computing: The Virtual Machine Introspection Approach," *JoWUA*, vol. 4, no. 1, pp. 135–152, 2013.

[14] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 1, pp. 11–33, Jan. 2004. [Online]. Available: http://dx.doi.org/10.1109/TDSC.2004.2

[15] Amazon EC2 Service Level Agreement. [Online]. Available: https://aws.amazon.com/ec2/sla/

[16] Z. Chen, F. Han, J. Cao, X. Jiang, and S. Chen, "Cloud Computing-Based Forensic Analysis for Collaborative Network Security Management System," *Tsinghua Science and Technology*, vol. 18, no. 1, pp. 40–50, Feb 2013.

[17] M. Mohite and S. Ardhapurkar, "Overcast: Developing Digital Forensic Tool in Cloud Computing Environment," in *Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on*, March 2015, pp. 1–4.

[18] F. Sharevski, "Digital Forensic Investigation in Cloud Computing Environment: Impact on Privacy," in *Systematic Approaches to Digital Forensic Engineering (SADFE), 2013 Eighth International Workshop on*, Nov 2013, pp. 1–6.

[19] K. Ruan, J. Carthy, T. Kechadi, and I. Baggili, "Cloud Forensics Definitions and Critical Criteria for Cloud Forensic Capability: An Overview of Survey Results," *Digital Investigation*, vol. 10, no. 1, pp. 34–43, 2013.

[20] S. Suneja, C. Isci, E. de Lara, and V. Bala, "Exploring VM Introspection: Techniques and Trade-offs," *SIGPLAN Not.*, vol. 50, no. 7, pp. 133–146, Mar. 2015.

[21] M. A. Law, "Using Net Present Value as a Decision-Making Tool," *Air Medical Journal*, vol. 23, no. 6, pp. 28 – 33, 2004.

[22] W. Wetekamp, "Net Present Value (NPV) as a tool supporting effective project management, year=2011," in *Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), 2011 IEEE 6th International Conference on*, vol. 2, Sept, pp. 898–900.

[23] K. Bernsmed, "Accountable Health Care Service Provisioning in the Cloud," in *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, Dec 2014, pp. 902–907.

[24] S. Vinoski, "CORBA: Integrating Diverse Applications within Distributed Heterogeneous Environments," *Communications Magazine, IEEE*, vol. 35, no. 2, pp. 46–55, Feb 1997.